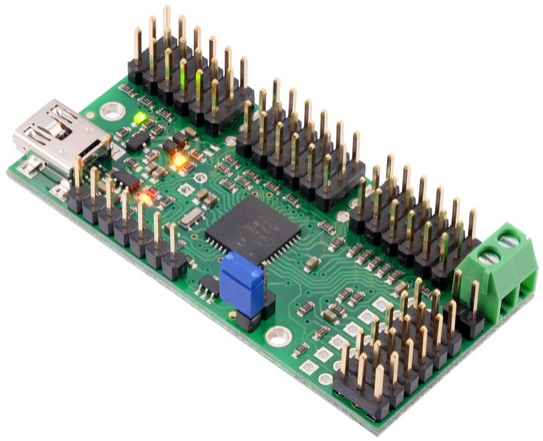
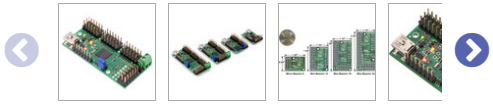
Mini Maestro 24-Channel USB Servo Controller (Assembled)





Pololu item #: 1356 **110** in stock
Brand: Pololu supply outlook
Status: Active and Preferred
✓RoHS 3
Free add-on shipping in USA
Free shipping in USA over $75

Price break	Unit price (US$)
1	49.95
5	45.95
25	42.28
100	38.90

Quantity: 1 **Add to cart**
backorders allowed **Add to wish list**

The 24-channel Mini Maestro 24 raises the performance bar for serial servo controllers with features such as a native USB interface and internal scripting control. Whether you want the best servo control available (0.25µs resolution with built-in speed and acceleration control and pulse rates up to 333 Hz) or a general I/O controller (e.g. to interface with a sensor or ESC via your USB port), this compact, versatile device will deliver. This fully-assembled version ships **with header pins installed**.

Alternatives available with variations in these parameter(s): channels partial kit? **Select variant...**

or or .

Description **Specs (10)** **Pictures (29)** **Resources (26)** **FAQs (6)** **On the blog (12)**

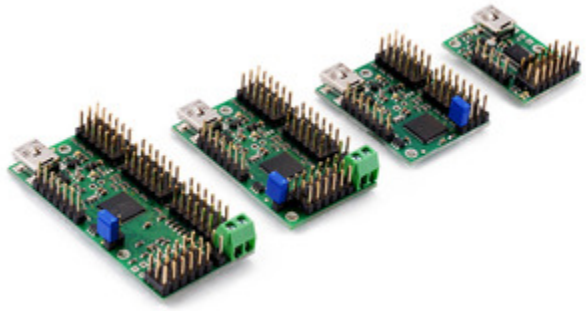
Getting started with the Maestro Servo Controller

For a full list of products shown in this video, see **the blog post**.

Overview

The Mini Maestros are the newest of Pololu's second-generation USB servo controllers, offering more channels and features than the smaller six-channel **Micro Maestro**. The Mini Maestros are available in three sizes, and they can be purchased fully assembled or as partial kits:

- **Mini Maestro 12 — fully assembled**
- **Mini Maestro 12 — partial kit**
- **Mini Maestro 18 — fully assembled**
- **Mini Maestro 18 — partial kit**
- **Mini Maestro 24 — fully assembled**

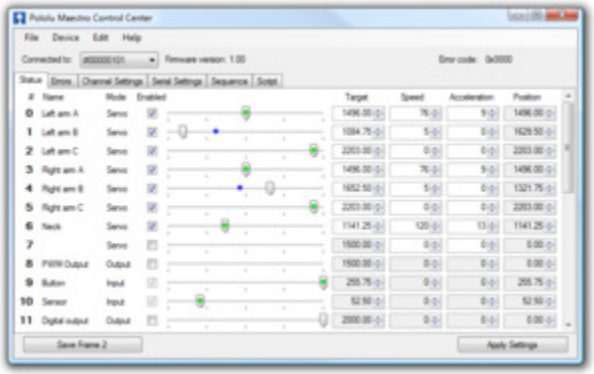


Maestro family of USB servo controllers: Mini 24, Mini 18, Mini 12, and Micro 6.

- **Mini Maestro 24 — partial kit**

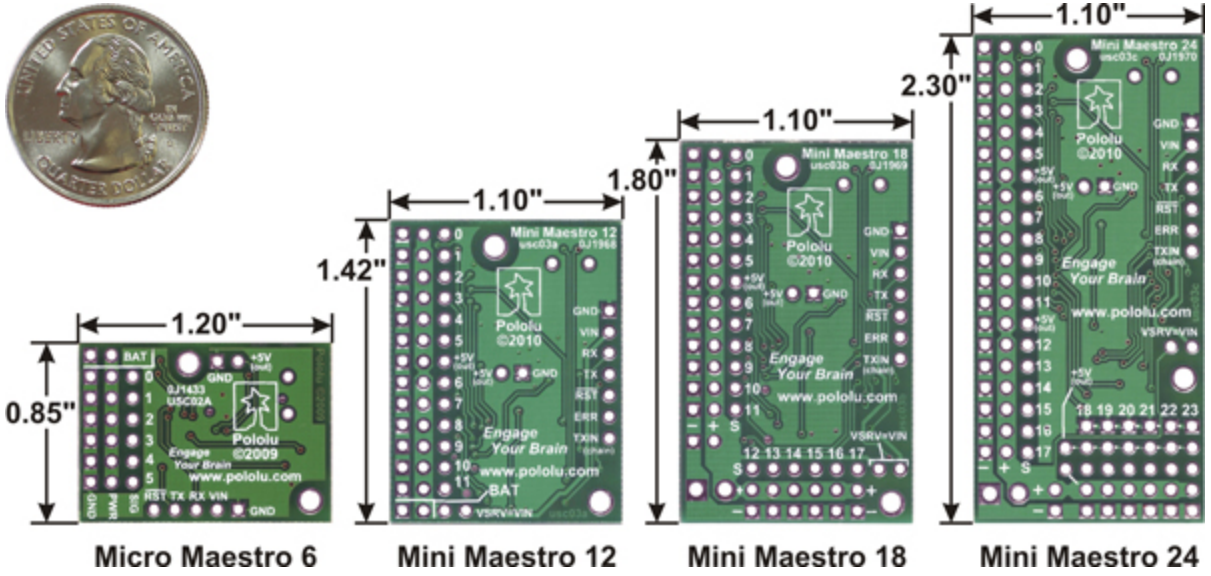
The Mini Maestros are highly versatile (and compact) servo controllers and general-purpose I/O boards. They support three control methods: USB for direct connection to a computer, TTL serial for use with embedded systems, and internal scripting for self-contained, host controller-free applications. The channels can be configured as servo outputs for use with **radio control (RC) servos** or electronic speed controls (ESCs), as digital outputs, or as analog/digital inputs. The extremely precise, high-resolution servo pulses have a jitter of less than 200 ns, making these servo controllers well suited for high-performance applications such as robotics and animatronics, and built-in speed and acceleration control for each channel make it easy to achieve smooth, seamless movements without requiring the control source to constantly compute and stream intermediate position updates to the Mini Maestros. The Mini Maestros also feature configurable pulse rates from 1 to 333 Hz and can generate a wide range of pulses, allowing maximum responsiveness and range from modern servos. Units can be daisy-chained with additional Pololu servo and motor controllers on a single serial line.

A free configuration and control program is available for Windows and Linux, making it simple to configure and test the device over USB, create sequences of servo movements for animatronics or walking robots, and write, step through, and run scripts stored in the servo controller. The Mini Maestros' 8 KB of internal script memory allows storage of up to approximately 3000 servo positions that can be automatically played back without any computer or external microcontroller connected.



The Status tab in the Maestro Control Center.

Because the Mini Maestros' channels can also be used as general-purpose digital outputs and analog or digital inputs, they provide an easy way to read sensors and control peripherals directly from a PC over USB, and these channels can be used with the scripting system to enable creation of self-contained animatronic displays that respond to external stimuli and trigger additional events beyond just moving servos.



The fully assembled versions of the Mini Maestro ship with **0.1″ male header pins** installed as shown in the respective product pictures. The partial kit versions ship with these header pins included but unsoldered, which allows the use of different gender connectors or wires to be soldered directly to the pads for lighter, more compact installations. A **USB A to mini-B cable** (not included) is required to connect this device to a computer. The Micro and Mini Maestros have 0.086″ diameter mounting holes that work with #2 and M2 screws.

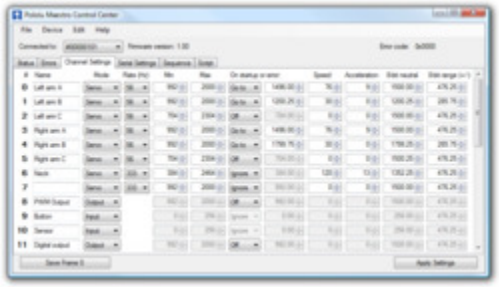
Main Features

- Three control methods: USB, TTL (5 V) serial, and internal scripting
- 0.25 µs output pulse width resolution (corresponds to approximately 0.025° for a typical servo, which is beyond what the servo could resolve)
- Pulse rate configurable from 1 Hz to 333 Hz [2]
- Wide pulse range of 64 µs to 4080 µs [2]
- Individual speed and acceleration control for each channel
- Channels can be optionally configured to go to a specified position or turn off on startup or error

Micro Maestro 6-channel USB servo controller (fully assembled) controlling three servos.

- Alternate channel functions allow the channels to be used as:
  - General-purpose digital outputs (0 V or 5 V)
  - Analog or digital inputs (channels 0 to 11 can be analog inputs; channels 12+ can be digital inputs)
  - One channel can be a PWM output with frequency from 2.93 kHz to 12 MHz and up to 10 bits of resolution
- A simple scripting language lets you program the controller to perform complex actions even after its USB and serial connections are removed
- Comprehensive **user's guide**

- Free configuration and control application for Windows and Linux makes it easy to:
  - Configure and test your controller
  - Create, run, and save sequences of servo movements for animatronics and walking robots
  - Write, step through, and run scripts stored in the servo controller
- Two ways to write software to control the Maestro from a PC:
  - Virtual COM port makes it easy to send serial commands from any development environment that supports serial communication
  - **Pololu USB Software Development Kit** allows use of more advanced native USB commands and includes example code in C#, Visual Basic .NET, and Visual C++
- TTL serial features:
  - Supports 300 bps to 200,000 bps in fixed-baud mode, 300 bps to 115,200 bps in autodetect-baud mode [(2)]
  - Simultaneously supports the Pololu protocol, which gives access to advanced functionality, and the simpler Scott Edwards MiniSSC II



The Channel Settings tab in the Maestro Control Center.

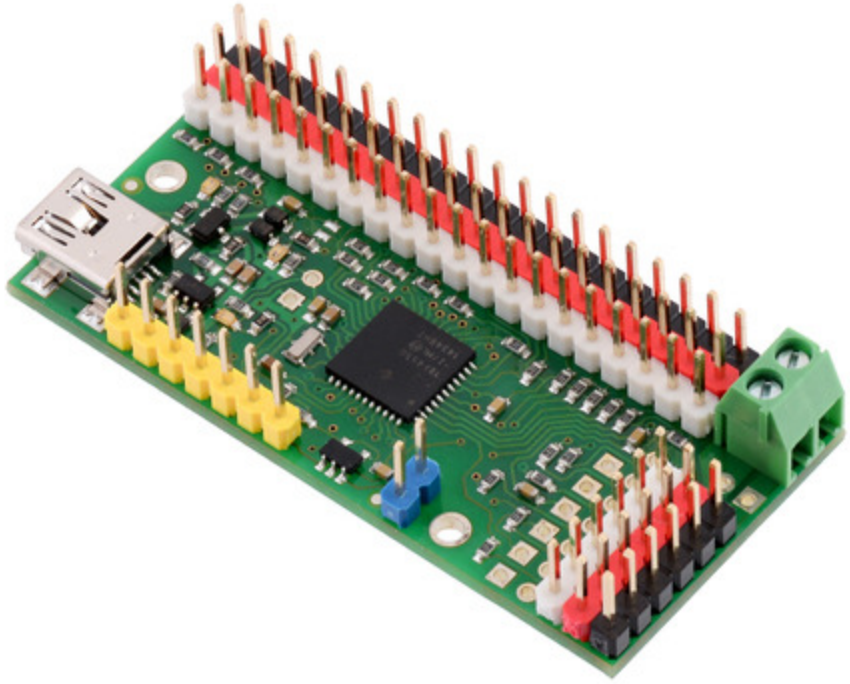
protocol (there is no need to configure the device for a particular protocol mode)
    - Can be daisy-chained with other Pololu servo and motor controllers using a single serial transmit line
    - Chain input allows reception of data from multiple Mini Maestros using a single serial receive line without extra components (does not apply to Micro Maestros)
    - Can function as a general-purpose USB-to-TTL serial adapter for projects controlled from a PC
- Our **Maestro Arduino library** makes it easier to get started controlling a Maestro from an **Arduino** or compatible boards like our **A-Stars**
- Board can be powered off of USB or a 5 V to 16 V battery, and it makes the regulated 5 V available to the user
- Upgradable firmware

Maestro Comparison Table

				
	Micro Maestro	**Mini Maestro 12**	**Mini Maestro 18**	**Mini Maestro 24**
Channels:	6	12	18	24
Analog input channels:	6	12	12	12
Digital input channels:	0	0	6	12
Width:	0.85" (2.16 cm)	1.10" (2.79 cm)	1.10" (2.79 cm)	1.10" (2.79 cm)
Length:	1.20" (3.05 cm)	1.42" (3.61 cm)	1.80" (4.57 cm)	2.30" (5.84 cm)
Weight(1):	3.0 g	4.2 g	4.9 g	6.0 g
Configurable pulse rate(2):	33 Hz to 100 Hz	1 Hz to 333 Hz	1 Hz to 333 Hz	1 Hz to 333 Hz
Pulse range(2):	64 µs to 3280 µs	64 µs to 4080 µs	64 µs to 4080 µs	64 µs to 4080 µs
Script size(3):	1 KB	8 KB	8 KB	8 KB
Price:	$24.95	$32.95	$41.95	$49.95

1 This is the weight of the board without header pins or terminal blocks.
2 The available pulse rate and range depend on each other and factors such as baud rate and number of channels used.
3 The user script system is more powerful on the Mini Maestro than on the Micro Maestro.

The Micro and Mini Maestros are available with through-hole connectors preinstalled or as partial kits, with the through-hole connectors included but not soldered in. The preassembled versions are appropriate for those who want to be able to use the product without having to solder anything or who are happy with the default connector configuration, while the partial kit versions enable the installation of custom connectors, such as **right-angle headers** that allow servos to be plugged in from the side rather than the top, or colored **header pins** that make it easier to tell which way to plug in the servo cables. The following picture shows an example of a partial-kit version of the 24-channel Mini Maestro assembled with colored male header pins:



24-channel Mini Maestro (partial kit version) assembled with colored male header pins.

Application Examples and Videos

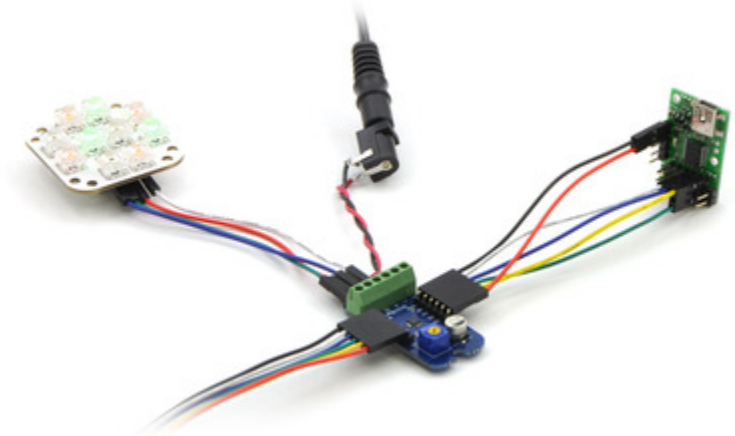
- Serial servo controller for multi-servo projects (e.g. robot arms, animatronics, fun-house displays) based on microcontroller boards such as the BASIC Stamp, **Orangutan robot controllers**, or Arduino platforms
- Computer-based servo control over USB port
- Computer interface for sensors and other electronics:
  - Read a **gyro or accelerometer** from a computer for novel user interfaces
- General I/O expansion for microcontroller projects
- Programmable, self-contained Halloween or Christmas display controller that responds to sensors

- Self-contained servo tester

Micro Maestro as the brains of a tiny hexapod robot.

An example setup using a **Micro Maestro** to control a **ShiftBar** and **Satellite LED Module** is shown in the picture below

and one of the videos above. Maestro source code to control a ShiftBar or **ShiftBrite** is available in the **Example scripts** section of the **Maestro User's guide**.



Connecting the Micro Maestro to a chain of ShiftBars. A single 12V supply powers all of the devices.

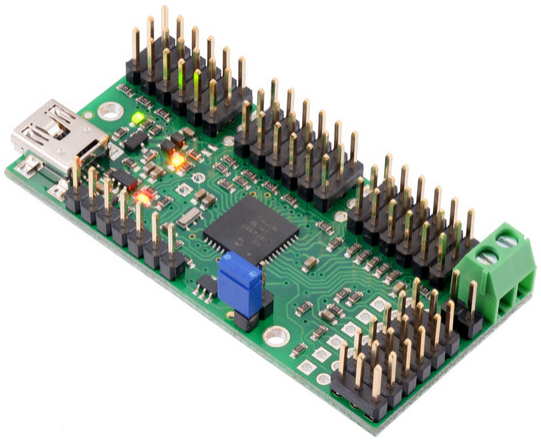
People often buy this product together with:

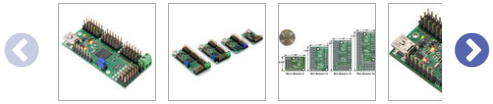
Micro Maestro 6-Channel USB Servo Controller (Assembled)

Mini Maestro 12-Channel USB Servo Controller (Assembled)

Mini Maestro 18-Channel USB Servo Controller (Assembled)

Mini Maestro 24-Channel USB Servo Controller (Assembled)





Pololu item #: 1356 110 in stock
Brand: Pololu supply outlook
Status: Active and Preferred
✓RoHS 3
Free add-on shipping in USA
Free shipping in USA over $75

Price break	Unit price (US$)
1	49.95
5	45.95
25	42.28
100	38.90

Quantity: 1 **Add to cart**
backorders allowed **Add to wish list**

The 24-channel Mini Maestro 24 raises the performance bar for serial servo controllers with features such as a native USB interface and internal scripting control. Whether you want the best servo control available (0.25µs resolution with built-in speed and acceleration control and pulse rates up to 333 Hz) or a general I/O controller (e.g. to interface with a sensor or ESC via your USB port), this compact, versatile device will deliver. This fully-assembled version ships **with header pins installed**.

Alternatives available with variations in these parameter(s): channels partial kit? **Select variant...**

or or .

Description **Specs (10)** **Pictures (29)** **Resources (26)** **FAQs (6)** **On the blog (12)**

Dimensions

Size:	1.10" x 2.30"
Weight:	12.2 g

General specifications

Channels:	24
Baud:	300 - 200000 bps[1]
Minimum operating voltage:	5 V
Maximum operating voltage:	16 V
Supply current:	40 mA[2]
Partial kit?:	N

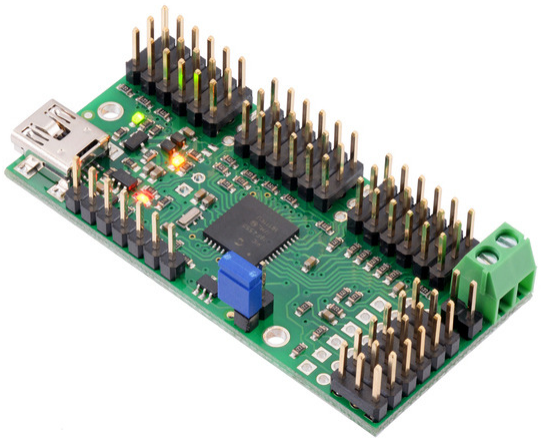
Identifying markings

PCB dev codes:	usc03c
Other PCB markings:	0J1970

Notes:

1 Autodetect works from 300 - 115200 bps.
2 With USB disconnected and all LEDs on. Connecting USB draws around 10 mA more.

Mini Maestro 24-Channel USB Servo Controller (Assembled)



Pololu item #: 1356 110 in stock
Brand: Pololu supply outlook
Status: Active and Preferred
✓RoHS 3
Free add-on shipping in USA
Free shipping in USA over $75

Price break	Unit price (US$)
1	49.95
5	45.95
25	42.28
100	38.90

Quantity: 1 **Add to cart**
backorders allowed **Add to wish list**

The 24-channel Mini Maestro 24 raises the performance bar for serial servo controllers with features such as a native USB interface and internal scripting control. Whether you want the best servo control available (0.25µs resolution with built-in speed and acceleration control and pulse rates up to 333 Hz) or a general I/O controller (e.g. to interface with a sensor or ESC via your USB port), this compact, versatile device will deliver. This fully-assembled version ships **with header pins installed**.

Alternatives available with variations in these parameter(s): channels partial kit? **Select variant...**

or or .

Description **Specs (10)** **Pictures (29)** **Resources (26)** **FAQs (6)** **On the blog (12)**

How should I attach a button or switch to my Micro Maestro?

First, decide which channel you would like to connect your button or switch to. In the Maestro Control Center, under the Channel Settings tab, change that channel to **Input** mode and click "Apply Settings". Next, wire a pull-up resistor (1–100 kilo-ohms) between the signal line of that channel and 5 V so that the input is high (5 V) when the switch is open. Wire the button or switch between the signal line and GND (0 V) so that when the button/switch is active the input will fall to 0 V. The picture to the right shows how to connect a button or switch to channel 0.

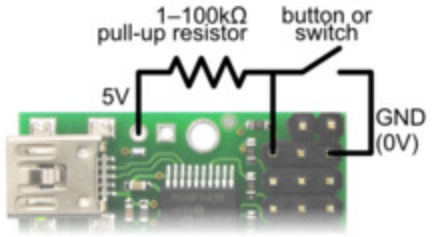
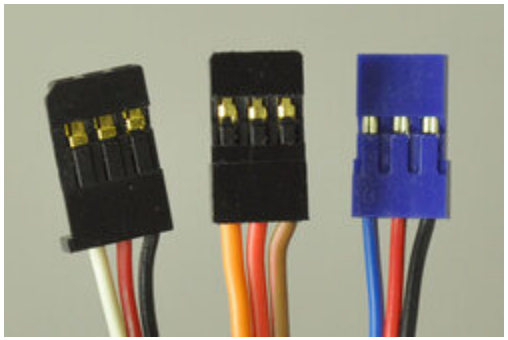


Diagram for connecting a button or switch to the Micro Maestro Servo Controller.

You can test your input by toggling the button/switch and verifying that the "Position" variable as shown in the Status tab of the Maestro Control Center reflects the state of your button/switch: it should be close to 255.75 when the button/switch is active and close to 0 when it is inactive. Now you can read the state of the button/switch in your script using the GET_POSITION command or over serial using the "Get Position" command. These commands will return values that are close to 1023 when the button/switch is active and close to 0 when it is inactive. **Warning: The Maestro's I/O lines can only tolerate voltages from 0 to 5 V, so if your power supply is more than 5 V be careful not to connect it to the signal line.**

What are the wires coming out of my servo?

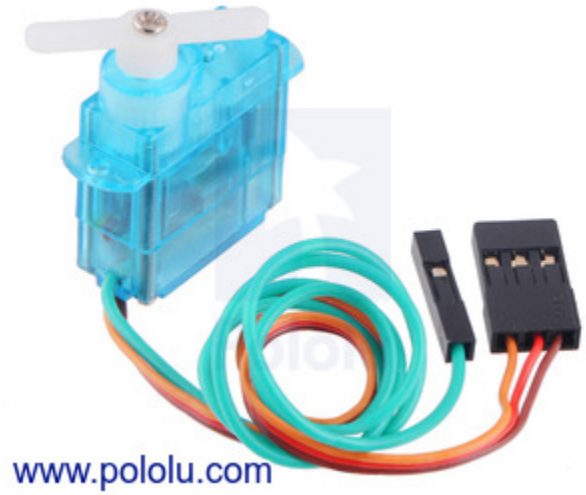
Most standard **radio control (RC) servos** have three wires, each a different color. Usually, they are either black, red, and white, or they are brown, red, and orange/yellow:



- brown or black = ground (GND, battery negative terminal)
- red = servo power (Vservo, battery positive terminal)
- orange, yellow, white, or blue = servo control signal line

Please check the specs for your servo to determine the proper power supply voltage, and please take care to plug the servo into your device in the proper orientation (plugging it in backwards could break the servo or your device).

Note: Some of the servos we carry also have an optional fourth green wire that is separate from the three standard ones. This wire provides access to the feedback potentiometer, allowing you to directly measure the position of the output. The servos with this extra wire have **"with Position Feedback"** at the ends of their product names. The picture below is an example of such a servo.



FEETECH Sub-Micro Servo FS0403-FB with Position Feedback.

Is it possible to use digital servos with the Maestro servo controllers?

Yes, any servo designed for standard RC receivers should work with the Maestro (and any of our other servo controllers), whether it is a digital or analog servo.

Can I power my servos with the USB port?

No. A USB port might only be capable of supplying 100 mA, which is less than what you need for a single servo. Many USB ports can deliver 500 mA, but this is still not enough for typical servos requiring 1 A or more.

How do I use my Maestro servo controller to get the maximum possible range of motion from my servo?

Be careful when going past the normal 90-degree range to avoid damaging your servo.

To find the settings in the Maestro Control Center that make your servo rotate as much as it can, first set the **Min** and **Max** values on the Channel Settings tab to a wider range. Then use the lowest possible supply voltage at which your servo moves and gradually move the slider on the status tab until the servo does not move any further or you hear the servo straining. Once you reach the limit, immediately move back from it to avoid damaging the servo. Finally, return to the channel settings tab and configure **Min** and **Max** so that the servo will never go past the limit.

Why do servo speed and acceleration limits not work for the first movement after startup or after setting the target to zero?

When the Maestro first starts up, the servos could be in any position, and the Maestro has no way of determining what position they are in. The standard RC servo protocol provides no way to get feedback from a servo. Therefore, when the Maestro receives its first Set Target command for a servo, whether it comes from serial, USB or an internal script, it will not be able to produce a smooth transition from the current position to the target position and will instead command the servo to immediately go to the target position. The speed and acceleration limits will work for subsequent commands since the Maestro will know where the servo should be and can produce servo pulses that smoothly change from the current position to the target position.

If you need your servo's first movement to be controlled by the speed and acceleration limits, then the first Set Target command you send to the Maestro should correspond to the servo's current position. For example, if you know that your servo will always be at a position of 1500 µs when your system starts up, then your first Set Target command for that servo should have a value of 1500 µs.

Similarly, if you set the target of a servo to zero to make the Maestro stop sending pulses, the Maestro will lose its knowledge of where the servo is. During this time, the servo might slip and go to a different position. If you know your servo is not going to slip, then your program or internal script could remember where the servo is and send a Set Target command with that position in it before trying to move the servo to another position.